Indexing Within — The Lost Gold Within the Enterprise

Brian Morrow Arthur Muir
Michael M. Gorlick*
Endeavors Technology
Irvine, California
bmorrow@endtech.com, amuir@endtech.com, mgorlick@endtech.com

August 22, 2000
Draft 1.1

Abstract

We outline a novel means to index *all* of the data on *every* computing device in an enterprise including devices well outside the reach of all currently available search engines. The invention resolves the problems of indexing intermittently connected devices, indexing large numbers of individual devices, and indexing device-specific content not normally considered amenable to indexing. Further the invention restricts indexing to authenticated search engines, enforces selective access to the information resolved by a query, and includes secure data transmission during indexing, query, and content inspection.

1 Introduction

Large scale web search requires a crawler, an indexer, and a query engine. *Crawlers*, as their name implies, "crawl" across a web following links and fetching web pages for the *indexer*, a massive compute- and storage-intensive application that constructs a comprehensive inverted index of every web page uncovered by the crawler. That index may contain significant words or phrases extracted from the content of the web page, metatags embedded within the HTML proper, and inferences made from the link structure of the page both outgoing and incoming.

The *query engine*, the application employed by end users, searches the indices constructed by the indexer to return links to candidate web pages in response to the query keywords and other criteria (language, domain of origin, age, ...) provided by the user. A *search engine* is a triumvirate of crawler, indexer, and query engine as illustrated in Figure 1. Web search is employed within, and by an enterprise, to assist employees, customers, and business partners in locating the enterprise information that they need.

Crawlers, like browsers, contact web servers for the pages that they are fetching and returning to the indexer. With rare exception these web servers are hosted on capacious machines with ample processing power and storage space. However, the number and form of computing devices within an enterprise is expanding. These devices vary in capacity, speed, platform, and method of network connection — ranging from corporate servers to laptops to personal digital assistants to embedded sensor and control networks. The domination of networks and the proliferation of devices tends to push data storage out to the edges of the corporate network yet the data and documents residing on these edge devices are often inaccessible to crawlers and indexing engines:

- No links refer to either the devices or their contents
- Some devices are only intermittently connected to the network
- The protocol assumed by the crawler (HTTP) is unsupported by the device
- The content of the device is in a format unknown to the indexing engine

Enterprises realize that much valuable and sensitive information resides in the devices at the edges of the network. and they attempt to capture that data with various mechanisms including shared file systems, enterprise-wide data repositories, knowledge management systems, and centralized data archives. Despite their widespread use these methods fail to capture much of the data on desktop, mobile, and personal devices.

Shared network volumes may capture only a fraction of the data on a desktop and the software required to support access is unsuitable for small, mobile devices. Data repositories

*Mailing Address: 19700 Fairchild, Suite 200, Irvine, CA 92612. Voice: 949.833.2800. Fax: 949.833.2881.

Diagram goes here!

Figure 1: The component parts of a search engine and their relationships.

frequently assume explicit submission; there is no guarantee that the repository contents are either current or comprehensive and the repository indexing (to the extent that it exists at all) may rely solely on keywords provided by the submitter. Knowledge management systems often rely on proprietary formats for content, restrict content to a small number of formats, or are specialized for a narrow domain, hence are ineffective for indexing and searching a broad array of corporate information sources. Often architected as centralized systems they are ill-suited for capturing the ephemera of network-centric mobile devices. Finally while corporate data archives are an effective means of retaining information of durable, lasting, and proven value they are are ineffectual in capturing the valuable, but short-lived data, created on desktop, portable, and mobile computing devices.

Sections 2–4 offer a solution to the problem of bringing previously inaccessible information into view of enterprise search engines. Section 2 presents a model of Web crawlers, one of the three fundamental components of modern web search engines. Section 3 introduces the Magi thin server technology and explains, using the taxonomy of Section 2, how the Magi server both extends the reach of search engines to devices that were until now inacessible and expands the flexibility, power, and timeliness of Web search engines. Finally, Section 4 presents (informally) the specific claims for our invention.

2 Taxonomy

Formally, a search engine is a triple (C, I, Q) of three separate, but related, applications — crawler, indexer, and query engine respectively [1].

Each may be characterized by action (what is done), locale (where it is done), and time (when it is done). In this manner a taxonomy of search engines can be constructed that characterizes the range of variation available to search engines component with respect to action, locale, and time. In the discussion below we consider only the crawler of the search engine. Future patents will address the the interactions between Magi servers and the indexer and query engine of a search engine.

2.1 Crawler

Crawlers fetch web pages for submission to the indexer of the search engine. Given an URL u a crawler C must first decide whether or not to visit the web page designated by u. If affirmative, the crawler can reach u if:

- C can connect to the host of u
- C has the authority to access the page designated by u

Connectivity and access authority are two separate, but related, considerations. The first is the ability to establish a TCP connection between the crawler and the web server and varies with the the position of the crawler within the network (for example relative to a firewall) or the network quality of service (such as congestion or routing anomolies).

Once a connection is established the crawler may be required to obtain permission to read the page (for example, it may be password protected). Access may also be restricted to a finite set of hosts whose identity is determined by inspecting the source IP address of the packet stream or by cryptographic means.

Finally, once the crawler has the page content in hand it must extract whatever links it can for the next round of crawling. Extraction depends on the form and semantics of the content and the extractors available to the crawler. For example, all crawlers can (by definition) extract links from HTML pages however, few crawlers have the extractors required to lift links embedded with PDF or Microsoft Word documents.

Thus a crawler is characterized by:

- The IP address of the crawler host which limits, with respect to network topology, routing, and firewalls, the remote hosts to which the crawler can connect
- Access authority
- A loading policy that gleans URLs of value from the set at hand
- An extraction policy that determines if the contents of a web page will yield URLs
- A set of extractors for extracting URLs from various forms of content

With this as background we present a formal model of a crawler that reflects the components and process outlined above. We adopt, with minor amendments, the definitions and notation of [2].

Notation 1 *Let an URL u be given, then p_u denotes the HTTP response for u (nominally the page contents of u). If U is a nonempty set of URLs then $p_U = \cup_{u \in U} p_u$.*

For the sake of conciseness take p_u to denote both the URL u and the page contents of u.

Definition 1 *An extractor is a function $g(p_u)$ such that either $g(p_u) = \perp$ meaning that g is ill-defined on p_u or $g(p_u) = S$ where S is the set (possibly empty) of all links (URLs) v extracted from p_u*

Definition 2 *A extraction policy $E(p_u)$ is a decision procedure that returns true if links (URLs) can be extracted from p_u and false otherwise.*

E may inspect the URL u, the MIME type of u (contained within the HTTP response) and the page contents p_u since all offer valuable hints as to the format and structure of p_u. For example, if the MIME type is "html" then p_u is a page whose structure is is well defined (by the HTML specification) and amenable to the extraction of links. If the MIME type is unspecified (the HTTP response omitted the `Content-Type` header field) then the crawler may examine the syntax of the URL or the content itself to infer the media type. For example, the URL suffix `.wav` or `.au` indicates (by common convention) an audio file which may contain links (rendered as speech) but whose extraction by machine agents is problematic at best. However, some audio formats make provision for the inclusion of digital metadata and the crawler, if equipped with a suitable extractor, may be able to extract that metadata for the benefit of the indexer.

Definition 3 *A loading policy is a decision procedure $L(u)$ that returns true if URL u is deemed suitable for loading and false otherwise.*

A page loading policy determines whether a crawler ignores robot excluded pages and generated pages (such as those produced by CGI scripts) and honors page loading, resource, and time limits with respect to a site or domain. Other considerations may also play a role in the formulation of L.

Definition 4 *Let an IP address α, a URL u, and a set of access permissions P (including passwords, encryption keys, and access procedures) be given. The access function $A(\alpha, u, P)$ returns p_u if and only if it is possible to access u from α and P grants sufficient authority.*

Definition 5 *A crawler C is a tuple $\langle \alpha, A, P, E, G, L \rangle$ where:*

- *α is the location (IP address) of C*
- *A is an access function*
- *P is a set of access permissions*
- *E is an extraction policy*
- *G is a nonempty set of extractors*
- *L is a loading policy*

Definition 6 *Let E a link extraction policy, an extractor g, and a nonempty seed set $S = \{u_0, \ldots, u_{m-1}\}$ of URLs be given. We say that an URL v can be extracted from p_S with respect to E and g if and only if there exists $u \in S$ such that*

- *$E(p_u)$ is true; and*
- *$v \in g(p_u)$*

Definition 7 *An URL u is reachable in one step by crawler $C = \langle \alpha, A, P, E, G, L \rangle$ with respect to a set of URLs S, written $S \overset{C}{\Rightarrow} u$, if $u \in S$ or:*

- *u can be extracted from p_S with respect to E and some $g \in G$*
- *$L(u)$ is true; and*
- *$A(\alpha, u, P) = p_u$*

Definition 8 *An URL u is reachable by crawler C with respect to a set of URLs S if $S \overset{C}{\Rightarrow}{}^{*} u$, that is, (S, u) is contained in the transitive closure of $\overset{C}{\Rightarrow}$.*

The definition of reachable is easily extended to page sets.

Definition 9 *Given page sets p_S, p_T then p_T is reachable from p_S by crawler C if and only if*

- *$p_S \subset p_T$; and*
- *$\forall p_t \in p_T \exists p_s \in p_S$ such that t is reachable by C from s*

The set of pages reachable by a crawler C is just the maximal page set in the relation $\overset{C}{\Rightarrow}{}^{*}$.

3 Magi Thin Server

Our solution expands the Internet search model to bring scaleable, comprehensive, and speedy enterprise-wide searching to any enterprise device. The cornerstone of this solution is the deployment of a specialized, resource-conservative, embedded HTTP server on every mobile, portable, and computer-enabled device within an enterprise.

Placing a small web server on each and every device enables existing search engines — without change or modification — to crawl, index, and search devices and information pools that were previously inaccessible. The Magi server is so small that it can be installed on virtually any device, including personal devices like the Palm Pilot, a WAP phone, or an embedded microcontroller, and it provides all of the services required by any Web compliant crawler.

Placing Magi on desktop workstations permits enterprise search engines to discover, index, and query the content resident on desktop machines that was, for all intents and purposes, unknown, uninventoried, and largely inaccessible. Using their existing search engine an enterprise can, with no additional effort, discover and access new sources of content enterprise-wide.

A device-specific Magi server translates device-dependent content into Web-standard formats such as HTML or XML. Consequently, information and data for which no extractors existed can now be crawled and extracted by the crawler of any common Web search engine. In effect the Magi server is supplying an extractor for the benefit of the crawler.

A Magi server can offload the task of extracting URLs and content from the crawler by creating virtual URLs that exist specifically for the benefit of a crawler. When a client (such as a crawler) visits such an URL (using the standard HTTP protocol so no changes to the crawler are required) the Magi server generates a dynamic page that summarizes the content of the original page. This reduces network transfers and load and improves the incisiveness of the indexing.

The technique of creating URLs for the benefit of a crawler can be further extended to create and push device-specific content to a repository or indexer. The Magi server also has the ability to authenticate and control access. Let u be an URL served by a Magi server M where u denotes content in a format for which no crawler extractor exists (for example, a device-specific configuration). M creates a virtual URL v that when accessed triggers the translation of the content of u into standard HTML. In this manner M performs extraction on behalf of the crawler giving it access to formats for which no crawler extractors exist.

In addition the architecture of the Magi server permits both device- and content-specific modules (plugins) that will translate non-HTML content into standard HTML for the benefit of a crawler. Finally, Magi can be configured as a crawler- and indexer-aware server that offers device- and content-dependent indexes directly to the crawler. In other words, the authentication protocols and access controls of Magi allow the Magi server to generate crawler-specific content that is optimized for the indexer of the search engine. For example, the Magi server executing on a Palm Pilot can generate a summary of the Pilot's memopad that is suitable for cross-indexing with a departmental project web site.

IThe notions of a crawler and indexer can be generalized considerably if the device server knows of, and cooperates with, the crawler. The architecture outlined here permits the deployment of enterprise- and domain-specific crawlers and indexers. Crawlers can be deployed within an enterprise to search for a specific form of content, for example, all content relating to a specific project. The crawler can move from device to device throughout the enterprise network and, with the cooperation of the Magi server onboard each device, can be served with just the content sought by the crawler, including relevant and incisive indices.

Intermittant connection is, at present, an insurmountable obstacle for crawlers as the target device (server) may not be connected to the network as the crawler is making its rounds. We offer several solutions to this problem; their suitability depends upon the capabilities of the target device:

- Magi servers announce their presence to the network when the device connects. This event notification is propagated to all interested subscribers including, for example, the enterprise crawler, thereby allowing the crawler to immediately visit the device and push needed content back to the enterprise indexer
- While offline Magi servers can preindex their relevant content for the search engine and when connected push an index set back to the search engine for inclusion in, and integration with, the enterprise index base
- Magi servers can push their content (all or selected portions) to the search engine (or to a proxy to deliver to the search engine on behalf of the device). This content is handed off to the indexer for index generation
- Alternatively, a Magi server can bleed its content incrementally to a search engine over the span of multiple connections to the network. This strategy might be appropriate for a device that is connected for only brief periods or supports only a low bandwidth connection

Because the Magi server on a device can act in an autonomous fashion it can index the device itself and notify the central index that the device is connected, and or has an update for the central index. In this fashion mobile and intermittently connected devices can be intelligently included in enterprise data searches. The search engine will not only be able to find data on all devices in the network, it will also instantly know the connection status of the device that contained the data pointed to by the metadata in the index. This opens the possibility of contacting the device owner in real-time to request that the device with critical data be connected as soon as possible. Magi can filter responses for sensitive data. This means that financial, HR, medical, personal, or other sensitive data can be contained appropriately. The power inherent in Magi authentication gives great control over how data is indexed and served, making enterprise-wide indexing a flexible yet secure tool.

4 Claims

Claim 1 *A system for indexing the:*

- *Information intake, contents, and output*
- *Hardware configuration, settings and status*
- *Software configuration, settings and status*
- *System and control logs*
- *Manner, rate, pattern, and frequency of use*

of any device with one or more embedded digital processors including, but not limited to, mobile phones, telephones, printers, fax machines, personal digital assistants, portable digital devices, digital media players and recorders, appliances, heating, ventilation, communication, and electrical systems, sensors and actuators, automotive electronic and mechanical systems, technical, scientific, and medical instruments, machine tools, and materiel handling, manufacturing, assembly, and delivery systems comprising:

- *For each such device a Magi web server resident on the device*
- *For each such device a network interface for fetching HTML pages from the devices in accordance with device-specific URLs and URL links*
- *For each such device one or more device-specific Magi modules to translate device-specific information into text, HTML, or XML web pages*
- *A Web search engine consisting of a crawler, indexer, and query engine*

Claim 2 *The system of claim 1 wherein devices whose network interface is disconnected from the network for extended periods or whose connection to the network is intermittent are crawled with content collected in order of decreasing priority or criticality.*

The presence service of Magi transmits a notification to all subscribers of the appearance (connection) of a Magi-enabled device. A subscriber may subscribe to presence notification for a given set of devices $d_1, \ldots, d_n$ or, with sufficient authority, the presence notification for all devices. In this manner the crawler can be notified when a specific device of interest connects to the network or when any device connects. When a device connects to the network (say, a personal digital assistant is set into its network and recharging cradle) the crawler is notified and immediately begin crawling the device, collecting pages from its resident Magi server. Crawling continues for as long as the device is connected. Note that the dynamic DNS service of Magi allows the crawler to obtain the IP address of the device even if it was dynamically assigned and changes from one network connect to another. Commercial crawlers, such as Ultraseek, provide an API for the implementation of crawling on demand.

A more selective form of crawling is possible. In this implementation the crawler waits for direct contact from the device itself in which the device informs the crawler of exactly where in the page space of the device to begin the crawl. In this manner the device can instruct the crawler to collect just those pages that have changed since the device was last visited by the crawler. The device-resident Magi server can alternatively inform the crawler of a set of pages to visit ordered by priority.

Finally, some search engines (Google, for example) archive the web pages collected by their crawlers. This facility allows a search engine user to view the web pages of a device even if it is disconnected since the search engine itself has a copy (albeit one that may be out-of-date).

Claim 3 *The system of claim 1 wherein devices whose network interface is of low bandwidth or unreliable are indexed.*

Low bandwidth connections present a particular challenge for crawling the contents of "bandwidth-challenged" devices. Here the device-resident Magi server can adopt tactics that ameliorate the deficiencies of the connection:

- Assuming that the Magi server and the crawler have a transport encoding in common the server can precompress offline the pages that it wants crawled and indexed. Using the mechanisms mentioned in Claim 2 it can direct the crawler to crawl just those pages for which it has precomputed compressed content. In this manner the device can make optimal use of its limited bandwidth
- The Magi server can break its content into small, individual "mini-pages", no one of which will require a substantial amount of transmission time. This technique, in combination with compression and directed crawling, permits incisive, directed "spot crawling" that allows a device to collaborate with a crawler even if the device is connected for only a brief period

Claim 4 *The system of claim 1 wherein only an authorized Web search engine may extract content from a device.*

Using the Magi security and authentication services the crawler can authoritatively identify itself to the device allowing the device to be crawled only by authorized crawlers.

Claim 5 *The system of claim 1 wherein personal, sensitive, or proprietary information is securely transferred from a device to an indexer.*

In addition the device and the crawler can securely establish a secret session key known to them alone that permits the secure transmission of sensitive information from the device to the crawler.

5 Implementation

Enterprise-wide indexing and search can be implemented in a gradual, stepwise fashion that allows the enterprise to gradually extend its reach to an ever broader range of devices adapting its crawling, indexing, and search services and need and circumstances dictate.

Beginning with its desktops an organization can install a Magi server on each personal workstation. Each such server is configured with a module that silently in the background, as processor cycles and disk bandwidth become available, indexes the relevant portions of the desktop file system. The desktop user need not schedule the server to create or update the index.

For a very small company without the resources for a sophisticated search system, each Magi server would simply serve up its metadata index on the fly as a query came in from an authorized (corporate) search engine. For larger organizations, or those desiring a high-performance search system, each Magi desktop server would push its metadata to a centralized index. The Magi server could be instructed to update the index at the desired level of granularity, or could be polled by the index server if desired. Intermittently connected devices would register their presence on the network and automatically update the index if their metadata has changed since the last connection. The Magi server can authenticate any connection to the device, hence there is no risk that data from a mobile device that is connected to a foreign (i.e. extra-enterprise) network will transmit inappropriate information to a non-authorized system. In a similar fashion the Magi server can filter access to data or metadata based on authentication circles.

6 Summary

Corporations will finally be able to realize the benefits of the data contained on the edge devices in their networks. As the Internet search engines have demonstrated, indexed searching makes it possible to quickly find data scattered over a huge number of rapidly changing devices. Using Magi technology, corporations can use proven, off-the-shelf, commodity components to quickly find any piece of data, on any device, in a secure fashion.

References

[1] S. Brin and L. Page, *The Anatomy of a Large-Scale Hypertextual Web Search Engine*, Proceedings of the Seventh International World Wide Web Conference, 1998, pp. 107–118.

[2] P. Bailey, N. Craswell, D. Hawking, *Chart of Darkness: Mapping a Large Intranet*, CSIRO CMIS Technical Report, Canberra, Austrailia, 2000.

[3] A. Rappaport, *Robots & Spiders & Crawlers: How Web and intranet search engines follow links to build indexes*, Search Tools Consulting, `www.searchtools.com`